SANDIA REPORT

SAND2001-1340
Unlimited Release
Printed May 2001

A Simple Plasma Code

Barry Marder

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/ordering.htm

SAND2001-1340
Unlimited Release
Printed May, 2001

A Simple Plasma Code

Barry Marder
Sandia National Laboratories
P. O. Box 5800
Albuquerque, NM 87185-1914

Abstract

As computers become faster, have more memory, and use multiple parallel processors, large, complex codes that more accurately simulate physical phenomena have emerged to utilize this capability. Most problems can benefit from this approach and many require it. But not all! There are problems for which simpler methods on more modest computers still work. The trick is to identify those problems, write the codes, and make their implementation sufficiently simple that they can be used conveniently by those who could profit from them. A Simple Plasma Code has been written with this philosophy in mind. It retains just enough physics to allow realistic simulations to be formulated and run quickly, even on a personal computer. This paper describes the physical model, its numerical implementation, and presents a sample simulation.

Contents

Figures

A Simple Plasma Code

I. Introduction

As computers become faster, have more memory, and use multiple parallel processors, large, complex codes that more accurately simulate physical phenomena have emerged to utilize this capability. Most problems can benefit from this approach and many require it. But not all! There are problems for which simpler methods on more modest computers still work. The trick is to identify those problems, write the codes, and make their implementation sufficiently simple that they can be used conveniently by those who could profit from them. This paper describes one such attempt.

II. The Physical Model

The Simple Plasma Code (SPC) represents an attempt to produce the simplest plasma simulation code that has enough physics to yield useful results for realistic problems. The code computes the steady-state behavior of a plasma consisting of electrons and ions in a region which could contain conductors and dielectrics. Ion motion is influenced solely by the electrostatic field:

$$\frac{d\mathbf{V}}{dt} = -\frac{Q}{M}\nabla\phi \tag{1}$$

where M is the ion mass, **V** is their velocity, ϕ is the electrostatic potential, and Q is the ion charge. Electron inertia is ignored by setting their mass equal to zero. Balancing the electron pressure with the electrostatic force gives:

$$\nabla(nkT) = qn\nabla\phi \quad \Rightarrow \quad n = \eta e^{\frac{q\phi}{kT}} \tag{2}$$

where n is the electron number density, q its (unsigned) charge, k is Boltzmann's constant, and T is the electron temperature, assumed to be constant. η is an integration constant.

The electrostatic potential is determined in one of two ways. Normally, it would be computed from Poisson's equation:

$$\nabla\bullet(\varepsilon\nabla\phi) = (qn - QN), \tag{3}$$

where N is the ion number density. In regions where the plasma is known to be essentially charge-neutral, ϕ is determined from (2) by equating the electron and ion charge densities.

For the numerical solution of (1) - (3), the equations are put into dimensionless form. Length is scaled by some characteristic dimension, L_0, ϕ by $\phi_0=(kT/q)$, velocity by $v_0=(kT/M)^{1/2}$, and number density is scaled by its value at injection, N_0. In dimensionless variables, the equations become

(4) $$\frac{d\mathbf{V}}{dt} = -\alpha\nabla\phi$$

(5) $$\nabla\bullet(\varepsilon\nabla\phi) = \beta(e^{\phi} - \alpha N),$$

where $\alpha = Q/q$ and $\beta = N_0q^2L^2/(\varepsilon_0kT) = (L/L_{Debye})^2$.

The numerical algorithm iterates between (4) and (5), alternately solving the former to generate enough ion trajectories to give the ion density, and then solving the latter to obtain a new potential from this density. The parameter η is determined from known values of potential and ion density in a charge-neutral region by setting n=N.

The parameter β can be very large. Where ϕ and its spatial derivatives are of order one, such as an enclosed low-voltage region, the ion and electron densities differ by order $1/\beta$. In this case, where the ion density is non-zero, ϕ can be approximated by imposing exact charge neutrality:

(6) $$\phi = \ln\left(\frac{\alpha N}{\eta}\right).$$

Where N=0, Poisson's equation, (5), is solved with only the electron density. Solving (6) is easier than solving (5), which involves the product of a very large number times a very small difference.

This is the extent of the physics in the model. Clearly, much has been omitted. The equations are non-relativistic and lack time dependence. There is no electron inertia and their temperature is assumed to be constant. Since there are no other ion species and no neutrals, there is neither ionization, charge exchange, nor recombination. No collision phenomena, such as viscosity, conductivity, heat conduction, or inter-species drag, are included. Nor is there a magnetic field. There are, obviously, problems for which this approach is totally inadequate, but for others, it can be useful. An example of one such case will be presented.

III. The Numerical Algorithm

The numerical algorithm begins with the creation of a rectangular grid, assumed, for illustration, to be two dimensional and Cartesian. The electrostatic potential and ion number density are specified on the grid.

The integration along an ion trajectory, defined by (4), is not performed with a time step, dt, but rather with a constant spatial step, ds, defined by

(7) $$ds^2 = dx^2 + dy^2 \quad \text{and} \quad \frac{dy}{dx} = \frac{V_y}{V_x}.$$

The size of ds is chosen to be less than either cell dimension, $ds < \min(\Delta x,\Delta y)$, so several steps are required to cross a cell. With this definition, (4) becomes

(8) $$|\mathbf{V}|\frac{d\mathbf{V}}{ds} = -\alpha\nabla\phi,$$

or, written in component form,

$$(9)\qquad |\mathbf{V}|\frac{dX}{ds} = V_x \qquad |\mathbf{V}|\frac{dY}{ds} = V_y \qquad |\mathbf{V}|\frac{dV_x}{ds} = -\alpha\frac{\partial\phi}{\partial x} \qquad |\mathbf{V}|\frac{dV_y}{ds} = -\alpha\frac{\partial\phi}{\partial y}.$$

Trajectories are initiated at designated locations and followed until they either exit the region or encounter an object, at which point they are terminated. The electric fields, the derivatives of ϕ, are defined the grid and interpolated to locations along trajectories. Each trajectory carries a flux, F, representing a number per unit time (F is scaled by $N_0L_0^2v_0$). At every step during the construction of a trajectory, an area-weighted density contribution is added to each corner of the surrounding cell. The amount deposited is the flux carried by the trajectory times the time it resides at that location, F dt = F ds/|**V**|. Summing the contributions from all trajectories and dividing by the volume of the cell gives the ion number density on the grid. Enough trajectories must be used to provide good statistics. When all trajectories have been created and the ion density determined, the potential consistent with this density is found from either (5) or (6).

Poisson's equation is solved using an exact method called a "Block Thomas Algorithm". This is an extension of a well known algorithm for solving two point boundary value problems. The 1-D version goes like this. Suppose we have a difference equation for ϕ of the form (m and n are used here as grid indices, not masses or densities):

$$(10)\qquad A_m\,\phi_{m+1} - B_m\,\phi_m + C_m\,\phi_{m-1} + D_m = 0$$

with boundary (Dirichlet or Neumann) conditions at m=1 and m=M. If we make the ansatz that the solution satisfies

$$(11)\qquad \phi_m = E_m\,\phi_{m+1} + F_m$$

substitution into (10) gives

$$(12)\qquad E_m = A_m/(B_m - C_mE_{m-1}) \quad \text{and} \quad F_m = (D_m + C_mF_{m-1})/(B_m - C_mE_{m-1}).$$

The solution is obtained by finding E_1 and F_1 from the boundary condition at m=1 and using (12) to find the remaining E's and F's by scanning forward. The other boundary condition gives ϕ_M (possibly using E_{M-1} and F_{M-1}) and the ϕ's are recovered by scanning (11) backwards. This double scan technique can be readily extended to two dimensions.

In the two dimensional algorithm, the scaler quantity ϕ_m becomes a vector of length N representing its values in the other direction. A, B, C and E each become MxN matrices while D and F become vectors of length M. Equations (11) and (12) have natural vector-matrix counterparts. For example, (12) becomes

$$(13)\qquad (B_m - C_mE_{m-1})\,E_m = A_m \quad \text{and} \quad (B_m - C_mE_{m-1})\,F_m = (D_m + C_mF_{m-1}).$$

This linear system is solved using LU decomposition. The boundary conditions in the scanned direction are found by extension of the 1-D method. Those in the other direction are obtained by suitably defining A, B, C, and D at the two boundaries, n = 1 and N.

If conductors at potential ϕ_k are in the domain, A=C=0, B=1, and D=ϕ_k at the designated locations. Similarly, in charge-neutral regions, (6) gives A=C=0, B=1, and D=ln(N/η).

The potential is obtained at each iteration with this technique. When a new density is found, the error in the difference equation is computed. The iteration stops when this error lies below some specified bound.

IV. An Example

To demonstrate the code, a simulation is presented of a compact device that accelerates an annular beam of singly ionized molecular deuterium. Figure 1 illustrates the simulation results. The underlying grid has 100 by 100 cells (101 points in each direction) and 5000 trajectories (not all of which are plotted).

The conductors in the source region are at zero potential, except for the emission block which is at +5 Volts. Those in the accelerator region are at -10 kV, except for the beam dump, which is at -9.5 kV. The relative dielectric constant of all dielectrics is 9. The electron temperature is 1 eV. A 0.25 amp ion current is injected from the top of the emission block with a directed and thermal velocity (spread) of 0.1 eV each. The computation is in (r,z) coordinates so an azimuthal thermal velocity component is included.

The potential in the source region drops about 5 Volts from where the ions are injected to the screen, as shown in figure 2. This electric field reflects the expanding ion beam. The beam, in turn, expands and accelerates because of the field. This "ambipolar" electric field arises to counter the hot electron's tendency to wander from the beam. Its magnitude is directly proportional to the electron temperature. If the electron temperature were doubled to 2 eV, the potential drop from source to screen would double to about 10 Volts and ions would reach the screen with twice the energy. At zero electron temperature, no electric field is required to keep the electrons bound to the ions and the beam would neither accelerate nor expand (except through its own temperature). The input file for this simulation is in the appendix.

V. Cheap Tricks

Coaxing a computer to yield realistic simulations in a timely manner often requires a bag of numerical tricks. Those used in the code, and in particular, for this example, are described here.

Cheap trick #1: Conductors and dielectrics are placed on the numerical grid as polygons and given either a potential or dielectric constant. A subroutine determines whether a point lies within a polygon by drawing a line from the point to "infinity" and counting the crossings with the boundary. An odd number of crossings means the point lies inside; an even number places it outside. The polygon tester places each grid point either in vacuum or inside an object and tags it accordingly. Note that the underlying grid is not conformal with these objects; they are simply overlaid on it. This limits resolution and care must be taken to insure that objects intersect enough grid points to define them and that they are not so narrow that a trajectory can pass through them undetected (the minimum width must be at least ds). As a trajectory is constructed, a check is made to determine if it has entered a polygonal object. If so, the trajectory is terminated. The test is performed only when the trajectory is in a cell that intersects an object.

Cheap trick #2: Cell volumes are used to compute density, but only that portion of a cell which lies in vacuum should be counted toward its volume because only it can contain ions. To compute cell volumes, all cells are loaded with a uniform array of trial "particles". For example, in the 5 x 5 array in figure 3, each particle will have $1/25^{th}$ of the cell's volume, $\Delta x \Delta y$. Particles that lie in an object are then removed. The effective cell volume is computed by summing the volumes of the remaining particles. This need be done only once.

Cheap trick #3: As already described, the potential in low voltage regions can be determined from the charge-neutral approximation, (6), when the ion density is not zero. This is not only computationally superior to using Poisson's equation to bludgeon the solution toward charge neutrality, it also eliminates plasma sheaths in those cases where they are not an important part of the solution. Sheaths arise at a conducting boundary because ϕ is given a prescribed value. If the grid does not have resolution on a Debye scale, non-physical gradients will occur as the difference equation attempts to satisfy this condition. The charge-neutral approximation eliminates the boundary condition associated with the partial differential equation, and with it the plasma sheath. This does not alter the solution in the bulk of the plasma since it would have been shielded from the conductor by the sheath. Of course, in problems where the sheath is to be computed, the charge-neutral approximation cannot be used.

Cheap trick #4: The solution is obtained iteratively, alternately finding the potential from the ion density, constructing the trajectories from potential gradients, and computing the ion density from the trajectories. It may, or may not converge, depending on the parameters. For example, higher current often causes convergence problems. There are ways to enhance convergence. In the charge-neutral approximation, the new potential consists of a fraction of the old potential, typically 50% but more if needed, combined with the newly computed one. This produces a damping effect, reduces overshoots, and encourages convergence. A similar technique is used in the Poisson solver, (10), by adding a term, W, to B and adding $W\phi_{old}$ to D. Determining the optimal expression for W is a formidable mathematical challenge, but experience can be used as a guide to find ones that usually work. In this example, W is set to some multiple of either $\beta(n+N)$ or $\beta(nN)^{1/2}$. The code has a restart capability, so a previous solution can be used as input for one with similar parameters. In general, however, the iteration converges reasonably quickly even when starting from scratch.

Cheap trick #5: A conducting screen separates the low-voltage source region, labeled I, from the high voltage ion acceleration region, labeled II. Because these regions are electrically isolated, a trajectory does not experience the high accelerating field until it has actually crossed the screen into region II. To ensure that this actually occurs, two separate potentials, ϕ_I and ϕ_{II}, are computed. For the first, all the high voltage potentials are set to zero, the screen is removed, and ϕ_I is found EVERYWHERE, using (6) where $N>0$ and (5) otherwise. The second potential, ϕ_{II} is set to zero in region I and solved in region II using Poisson's equation with the screen and the correct potentials. As a trajectory develops, the polygon tester determines its region. If it is in region I, it responds to ϕ_I. If it is in region II, it responds to ϕ_{II}. Thus, the resolution is better than might be expected and the actual shape of the screen is incorporated with minimal imprinting from the underlying rectangular grid. The two potentials are obtained sequentially, because there should be no feedback from region II to region I. (The algorithm fails if trajectories are turned around in region II and re-enter region I.) To account for the screen's transparency, a trajectory's flux is reduced by the specified amount as it crosses the screen.

IV. Summary

Approximations and omissions are nearly always made when describing physical systems. If a system is to be simulated on a computer, additional numerical approximations are usually necessary. It is the scientist's job to decide what should be kept and what can be neglected or approximated. Having done this, the simplest numerical model should be used. This paper presents one such approach for a limited, but useful, class of problems.

Appendix

The input file for the ion accelerator example

```
'SPC example input file'
'r-grid, z-grid, number of trajectories, initial iterations'
 101  101   5000   10

'Maximum error, numerical convergence parameters. 0<c1<1, 0<c2'
 0.01    0.5     1.

'Number of trajectories plotted'
 40

'Enter 0 for lengths in centimeters, 1 for inches'
 0

'Edirected, Ethermal, Te(eV), average atomic ion mass, ion charge'
 0.1    0.1     1.      4.      1.

'Injected ion current(A), Emitter voltage(V), Remit, Zemit, screen transparancy'
 0.25    5.    0.1    0.1     0.5

'Number of conductor and dielectric polygonal elements'
 8    3

'Descriptive name of conductor'
'Number of vertices. Potential'
'List of vertices (r,z) of conductor polygons'

'Emission block (must have anode potential)'
 4    5.
 0.         ..0
 .100     .0
 .100     .100
 0.         100

'Inner screen support'
 4    0.
 0.         .500
 .270     .500
 .270     .550
 0.         .550

'Outer screen support with flange'
 6     0.
 .450     0.
```

```
1.10     0.
1.10     .100
.500     .100
.500     .360
.450     .360

'Inner high voltage electrode'
6    -1.e4
0.       1.10
.450     1.10
.450     1.20
.050     1.20
.050     1.75
0.       1.75

'Outer high voltage electrode'
6    -1.e4
1.20     1.10
1.80     1.10
1.80     1.75
1.75     1.75
1.75     1.20
1.20     1.20

'Far wall'
4    -1.e4
0.       1.75
1.80     1.75
1.80     1.85
0.       1.85

'Surrounding case'
6     -1.e4
1.95     0.
2.00     0.
2.00     2.00
0.       2.00
0.       1.95
1.95     1.95

'Beam dump'
4    -0.95e4
.500     1.55
1.50     1.55
1.50     1.65
.500     1.65
```

```
'Descriptive name of insulator'
'Number of vertices. Relative dielectric constant'
'List of vertices (r,z) of dielectric polygons'

'Beam cutoff'
 8     9.
 .100     0.
 .450     0.
 .450     .220
 .200     .220
 .200     .170
 .380     .170
 .380     .100
 .100     .100

'Enclosing dielectric'
 8     9.
 1.10     0.
 1.95     0.
 1.95     1.95
 0.       1.95
 0.       1.85
 1.80     1.85
 1.80     1.10
 1.10     .100

'Beam dump holder'
 4     9.
 .700     1.65
 1.30     1.65
 1.30     1.75
 .700     1.75

'Number of vertices in charge neutral region. '
'List of vertices (r,z) of polygon enclosing charge-neutral region'
 6
 0.       0.
 .500     0.
 .500     .360
 .450     .360
 .270     .550
 0.       .550
```

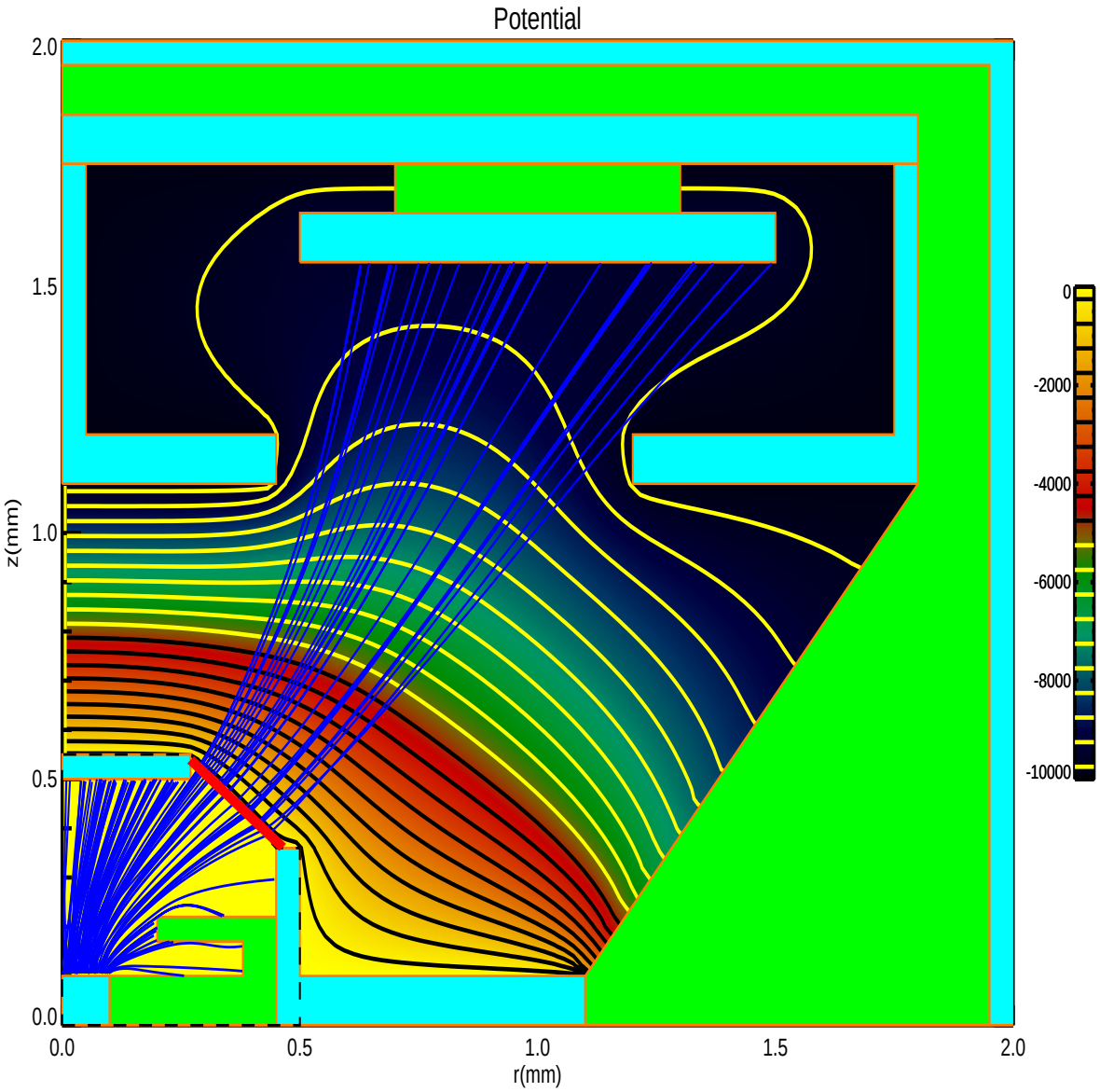

Figure 1. Potential contours and ion trajectories for an ion accelerator. (Blue=conductors, green=dielectrics). The center line is the left vertical axis. A 50% transmissive screen (red) separates the source region from the accelerator region.

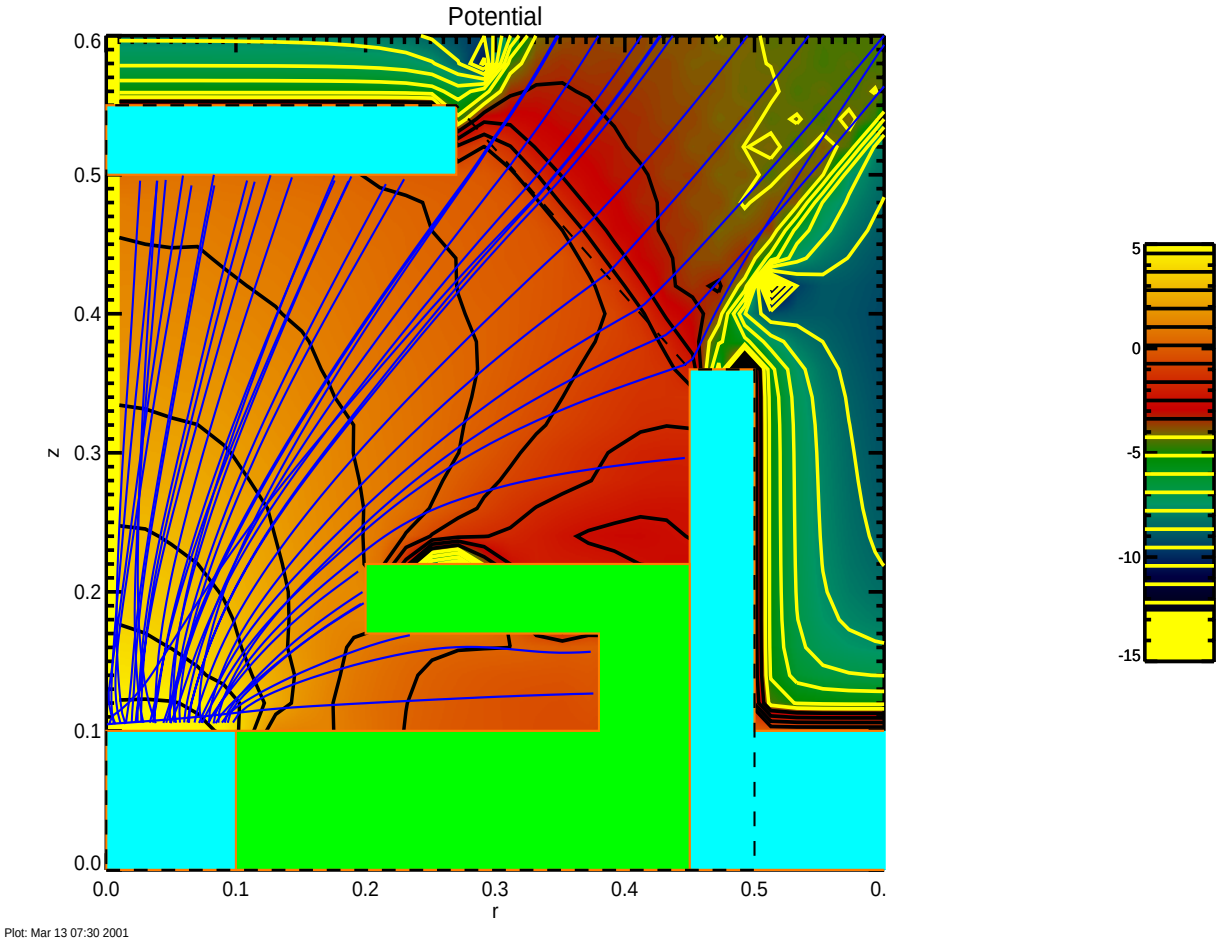

Figure 2. The potential, ϕ_I, and trajectories in the source region, obtained from the charge-neutral approximation where N>0 and from Poisson's equation where N=0.

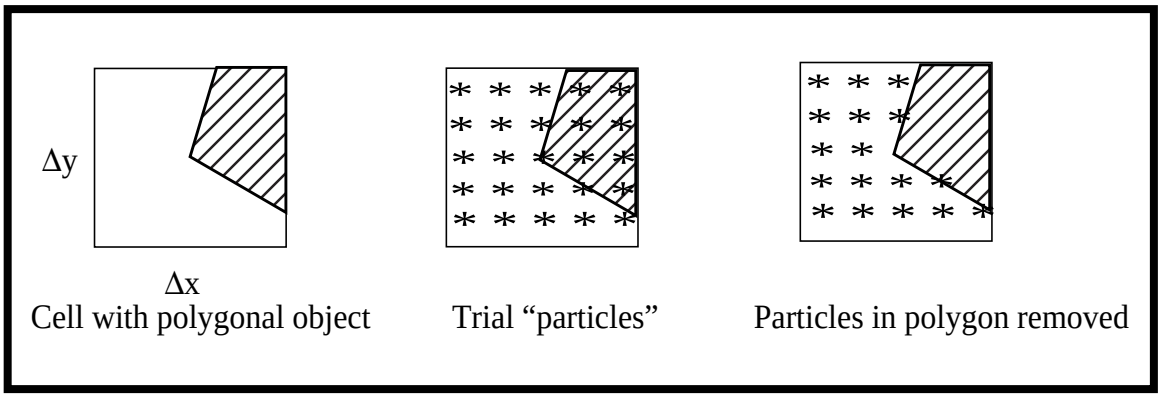

Figure 3. Determining the vacuum volume of a cell by removing test particles in a polygonal object and summing the volume of those remaining.

Distribution

1	MS 1056	Bill Wampler, 1111
1	MS 1423	Merle Riley, 1118
1	MS 1423	Greg Hebner, 1118
1	MS 1423	Paul Miller, 1118
1	MS 1186	Tom Mehlhorn, 1674
1	MS 0521	Frank Bacon, 2502
1	MS 0516	Gary Laughlin, 2564
1	MS 0516	Bud Burns, 2564
1	MS 0516	John Brainard, 2564
1	MS 0516	Carla Busick, 2564
1	MS 0516	Robert Koss, 2564
1	MS 0827	Tim Bartel, 9113
1	MS 0367	Rick Buss, 9117
1	MS 0834	Matthew Hopkins, 9117
1	MS0836	John Torczynski, 9117
1	MS 0836	Bob Campbell, 9117
1	MS 0836	Richard Griffith, 9117
1	MS 0836	Ann Moats, 9117
1	MRC	Dale Welch
10	MS 1186	Barry Marder, 1674
1	MS9018	Central Technical Files, 8945-1
2	MS0899	Technical Library, 9616
1	MS0612	Review and Approval Desk for DOE/OSTI, 9612